Estimation of General Curves and Surfaces to Edge and Range Data by Euclidean Fitting

Division of Informatics, University of Edinburgh

Institute of Perception, Action and Behaviour

Estimation of General Curves and Surfaces to Edge and Range Data by Euclidean Fitting

by

Petko Faber, Robert Fisher

Informatics Research Report EDI-INF-RR-0146

Division of Informatics **August 2002**
http://www.informatics.ed.ac.uk/

Estimation of General Curves and Surfaces to Edge and Range Data by Euclidean Fitting

Petko Faber, Robert Fisher

Informatics Research Report EDI-INF-RR-0146

DIVISION *of* INFORMATICS
Institute of Perception, Action and Behaviour

August 2002

Abstract :

The ability to construct CAD or other object models from edge and range data has a fundamental meaning in building a recognition and positioning system. While the problem of model fitting has been successfully addressed, the problem of efficient high accuracy and stability of the fitting is still an open problem. This paper addresses the problem of estimation of general curves and surfaces to edge and range data by a constrained Euclidean fitting. We study and compare the performance of various fitting algorithms in terms of efficiency, correctness, robustness and pose invariance, and present our results improving the known fitting methods by an (iterative) estimation of the real Euclidean distance.

Keywords : Euclidean distance, approximate distance, curve fitting, surface fitting, M-estimator

Division of Informatics, University of Edinburgh

Institute for Perception, Action and Behaviour

Estimation of General Curves and Surfaces to Edge and Range Data by Euclidean Fitting

by

Petko Faber and Robert B. Fisher

Informatics Research Report EDI-INF-RR-0146

Division of Informatics **August 2002**
http://www.informatics.ed.ac.uk/

Estimation of General Curves and Surfaces to Edge and Range Data by Euclidean Fitting

Petko Faber and Robert B. Fisher

Informatics Research Report EDI-INF-RR-0146

DIVISION *of* INFORMATICS
Institute for Perception, Action and Behaviour

August 2002

Abstract : *The ability to construct CAD or other object models from edge and range data has a fundamental meaning in building a recognition and positioning system. While the problem of model fitting has been successful addressed, the problem of efficient high accuracy and stability of the fitting is still an open problem. This paper address the problem of estimation of general curves and surfaces to edge and range data by a constrained Euclidean fitting. We study and compare the performance of various fitting algorithms in terms of efficiency, correctness, robustness and pose invariance, and present our results improving the known fitting methods by an (iterative) estimation of the real Euclidean distance.*

Keywords : Euclidean distance, approximate distance, curve fitting, surface fitting, M-estimator.

1 Introduction

Shape analysis of objects is a key problem in computer vision with several important applications in manufacturing, such as quality control and reverse engineering. However, the application of shape in computer vision has been limited to date by the difficulties in its computation. To build a recognition and positioning system based on implicit curves and surfaces it is imperative to solve the problem of how curves and surfaces can be fitted to the data extracted from single or multiple 3D images. The fitting process is necessary for automatically constructing object models and for building intermediate representations from observations during the recognition. Implicit polynomial curves and surfaces are potentially among the most useful object or data representations for use in computer vision and image analysis. Their power appears by their ability to smooth noisy data, to interpolate through sparse or missing data, their compactness and their form being commonly used in numerous constructions.

Let $f(\vec{x})$ be an *implicit polynomial* of degree d given by

$$f(\vec{x}) = \sum_{\substack{(i+j+k)\leq d \\ \{i,j,k\}\geq 0}} a_{ijk} \cdot x^i \cdot y^j \cdot z^k = 0 \ . \tag{1}$$

Then, we only have to determine the parameter set $\{a_{ijk}\}$ which describes the given data best. Parameter estimation is usually cast as an optimization problem, which can be solved in many ways because of different optimization criteria and several possible parameterizations. Generally, the literature on fitting can be divided into three general techniques: least-squares fitting (e.g. [Allen 1935, Bookstein 1979, Fitzgibbon and Fisher 1995, Kanatani 1993, Rosin 1993]), Kalman filtering (e.g. [Chui and Chen 1987, Degeeter *et al.* 1997, Dickmanns and Graefe 1988]), and robust clustering techniques (e.g. [Besl and Jain 1985, Duda and Haet 1972]). While the clustering methods are based on mapping data points to the parameter space, such as the Hough transform and the accumulation methods, the least-squares methods are centered on finding the sets of parameters that minimize some distance measures between the data points and the curves and surfaces. Unfortunately, the estimation of the Euclidean distances from a data point to a *general* curve and surface has been sometimes computationally impractical, because for some cases there is no closed form expression for the Euclidean distance, and approximations or iterative methods are required to estimate it. For approximation, often the result of evaluating the characteristic polynomial $f(\vec{x})$ is taken, or the first order approximation, suggested by Taubin [Taubin 1991], is used. However, experiments with the Euclidean distance show the limitations of approximations regarding quality and accuracy of the fitting results. Additionally, when using an approximation, the invariance of the fitting to Euclidean transformations is not guaranteed. It is obvious that accuracy and stability of the fitting has a substantial impact on recognition performance especially in reverse engineering where we desire an accurate reconstruction of 3D geometric models of objects from range data. Thus it is very important to get good shape estimates from the data.

In this paper we

1. summarize how to compute the Euclidean distance function for common surfaces and general quadric surfaces,
2. summarize a little-known Euclidean distance function to the ellipse,
3. give an efficient algorithm for least-squares fitting using the Euclidean distance, and
4. compare this fitting to three other main fitting distance measures and conclude that the Euclidean distance is much more accurate and stable without extraordinary computational expense.

2 Fitting of general curves and surfaces

An implicit curve or surface is the zero set of a smooth function $f : \mathbb{R}^n \to \mathbb{R}^m$ of the n variables: $\mathcal{Z}(f) = \{\vec{x} : f(\vec{x}) = 0\}$. In our applications we are interested in three special cases: $\mathcal{Z}(f)$ is a *planar curve* if $n = 2$ and $m = 1$, it is a *surface* if $n = 3$ and $m = 1$ and it is a *space curve* if $n = 3$ and $m = 2$. Given a finite set of data points $\mathcal{D} = \{\vec{x}_p\}$, $p \in [1, P]$, the problem of fitting a general curve and surface $\mathcal{Z}(f)$ to $\mathcal{D}$ is usually cast as minimizing a distance measure

$$\frac{1}{P}\sum_{p=1}^{P} \mathrm{dist}\left(\vec{x}_p, \mathcal{Z}(f)\right) \to \ \mathrm{Min} \tag{2}$$

from the data points to the curve or surface $\mathcal{Z}(f)$, a function of the set of parameters $\{a_{ijk}\}$ of the polynomial. The distance from the point $\vec{x}_p$ to the zero set $\mathcal{Z}(f)$ is defined as the minimum of the distances from $\vec{x}_p$ to points $\vec{x}_t$ in the zero set $\mathcal{Z}(f)$:

$$\mathrm{dist}(\vec{x}_p, \mathcal{Z}(f)) = \min\left\{\| \vec{x}_p - \vec{x}_t \| : f(\vec{x}_t) = 0\right\} \ . \tag{3}$$

In the past researchers have often replaced the real Euclidean distance by an approximation because of computational efficiency. But it is well known that a) a different performance function can produce a very biased result, and b) for a lot of primitive curves and surfaces like straight lines, ellipses, planes, cylinders, and cones, there is a closed form expression for the Euclidean distance from a point to the zero set.

In the following we summarize the Algebraic fitting, the 3L fitting suggested by Blane et.al. [Blane *et al.* 2000], a fitting method suggested by Taubin [Taubin 1991, Taubin 1993] and our Euclidean fitting method.

2.1 Algebraic fitting (*AF*)

Algebraic fitting is based on the approximation of the Euclidean distance between a point and the curve or surface by the algebraic distance $\mathrm{dist}_A\left(\vec{x}_p, \mathcal{Z}(f)\right) = f(\vec{x}_p)$. Given the algebraic distance for each point, Eq. (2) can be formulated as an eigenvector problem

$$\vec{D}^T\vec{D} \cdot \{a_{ijk}\} = \vec{\lambda} \cdot \{a_{ijk}\} \tag{4}$$

where $\vec{D}$ is the matrix of monomials for the data set

$$\vec{D} = \left[\left\{x_p^i \cdot y_p^j \cdot z_p^k\right\}_{\substack{(i+j+k)\le d \\ \{i,j,k\}\ge 0}}\right]_{p=1,\ldots,P} . \tag{5}$$

To avoid the trivial solution $\{a_{ijk}\} = \vec{0}$ and any multiple of a solution, the parameter vector $\{a_{ijk}\}$ may be constrained in some way (e.g. [Albano 1974, Bookstein 1979, Fitzgibbon and Fisher 1995, Paton 1970]) which leads to the generalized eigenvector problem

$$\vec{D}^T\vec{D} \cdot \{a_{ijk}\} = \vec{\lambda} \cdot \vec{C} \cdot \{a_{ijk}\} \ . \tag{6}$$

The most used constraints are either *linear*, $\mathrm{diag}(\vec{C})\cdot\{a_{ijk}\} = 1$, or *quadratic*, $\{a_{ijk}\}^T \cdot \vec{C} \cdot \{a_{ijk}\} = 1$. The solution gives a set of eigenvalues $\vec{\lambda}$, the solution is chosen that yields to the lowest residual. The corresponding eigenvector is the parameter vector $\{a_{ijk}\}$.

The pros and cons of using algebraic distances are a) the gain in computational efficiency, because closed form solutions can usually be obtained, on the one hand and b) the often unsatisfactory results on the other hand.

2.2 3L fitting (*LF*)

The 3L fitting method is based on the algebraic distance too, but it is slightly different. 3L fitting uses, besides the original data set, a pair of synthetically generated data sets consisting of points at a distance c to either side of the original data set. This pair can be computed by a distance transform. The 3L fitting itself is formalized as a linear least-squares explicit polynomial fitting problem. The solution for the parameter vector $\{a_{ijk}\}$ can be obtained immediately by

$$\{a_{ijk}\} = M_{3L}^{\div} \cdot \vec{b}\,, \tag{7}$$

where $\vec{M}_{3L}^{\div} = (\vec{M}_{3L}^T \vec{M}_{3L})^{-1} \vec{M}_{3L}^T$ denotes the pseudoinverse of $\vec{M}_{3L}$. The block matrix M_{3L} and the block column vector $\vec{b}$ are defined as

$$\vec{M}_{3L} = \begin{bmatrix} \vec{M}_{\Gamma_{-c}} \\ \vec{M}_{\Gamma_0} \\ \vec{M}_{\Gamma_{+c}} \end{bmatrix} \qquad \vec{b} = \begin{bmatrix} -\vec{c} \\ \vec{0} \\ +\vec{c} \end{bmatrix} \tag{8}$$

where $\vec{M}_{\Gamma_0}$, $\vec{M}_{\Gamma_{-c}}$, and $\vec{M}_{\Gamma_{+c}}$ are the matrices of monomials for the original data set respectively the synthetically generated data sets. The values $\pm c$ is the (Euclidean) distance of $\vec{M}_{\Gamma_{-\pm}}$ to $\vec{M}_{\Gamma_0}$. Although the 3L fitting is computationally simple and fast, the cost of generating the sets Γ_{-c} and Γ_{+c} might be quite substantial because of the need of 3D morphological transformations. The pros and cons of introducing the additional data sets are the improved stability and robustness in the presence of noisy or missed data on the one hand, but on the other hand the inferior accuracy compared with the other summarized methods. For a detailed description on how to estimate the parameter set out of the three data sets see [Blane *et al.* 2000].

2.3 Taubin's fitting (*TF*)

An alternative to approximately solve the minimization problem is to replace the Euclidean distance from a point to an implicit curve or surface by the first order approximation [Taubin 1991].

$$\mathrm{dist}_T\left(\vec{x}_p, \mathcal{Z}(f)\right) = \frac{|f(\vec{x}_p)|}{\|\nabla f(\vec{x}_p)\|} \tag{9}$$

Restating the problem as the minimization this distance measure, the minimizer is then the eigenvector of the generalized eigensystem

$$\vec{D}^T \vec{D} \cdot \{a_{ijk}\} = \vec{\lambda} \cdot \vec{J_D}^T \vec{J_D} \cdot \{a_{ijk}\} \tag{10}$$

where $\vec{J_D}$ is the Jacobian matrix of the matrix $\vec{D}$ of monomials for the data set.

Besides the fact that no iterative procedures are required, the fundamental property is that it is a first order approximation to the exact distance. But, it is important to note that the approximate distance is also biased in some sense. If, for instance, a data point $\vec{x}_p$ is close to a critical point of the polynomial, i.e., $\|\nabla f(\vec{x}_p)\| \approx 0$, but $f(\vec{x}_p) \neq 0$, the distance becomes large. This is certainly a limitation. To minimize Eq. (2) the usage of the Levenberg-Marquardt (*LM*) algorithm [Levenberg 1944, Marquardt 1963] is suggested by G. Taubin [Taubin 1993].

2.4 Euclidean fitting (*EF*)

To overcome the problems with the approximate distance metrics, it is natural to use instead the Euclidean distance, which is invariant to transformations in Euclidean space and is not biased. In Sec. 3 we argue how the Euclidean distance can be estimated between a point and the zero set $\mathcal{Z}(f)$. Given the Euclidean distance $\mathrm{dist}_E(\vec{x}_p, \mathcal{Z}(f))$ for each point the following simple algorithm can be used:

1. The Euclidean fitting requires an initial estimate for the parameters $\{a_{ijk}\}$ and we have found (cf. Sec. 4) that the results of Taubin's fitting method are better than the others. This gives the initial parameter set $\{a_{ijk}\}^{[0]}$.

2. In the second step the estimated parameter are updated using a nonlinear optimization algorithm $\{a_{ijk}\}^{[s+1]} = F_{LM}(\{a_{ijk}\}^{[s]})$. Our implementation is based on the *LM* algorithm which has become the standard of nonlinear optimization routines. It combines the inherent stability of the Steepest Gradient Descent with the quadratic convergence rate of the Gauss-Newton method.

3. The new estimates $\{a_{ijk}\}^{[s+1]}$ are evaluated by a so-called *M-estimator* $\mathcal{L}$ on the basis of $\mathrm{dist}_E(\vec{x}_p, \mathcal{Z}(f))$. If $\mathcal{L}(\{a_{ijk}\}^{[s+1]}) < \mathcal{L}(\{a_{ijk}\}^{[s]})$ the $\{a_{ijk}\}^{[s+1]}$ will be accepted and the fitting will be continued with step 2. Otherwise the fitting is terminated and $\{a_{ijk}\}^{[s]}$ is the desired solution. In Sec. 3.3 we discuss the problem of selecting a suitable error function $\mathcal{L}$.

2.5 Summary

Combining Eq. (2) (to minimizes the distance error of all points) and Eq. (3) (to get the distance error of one point) to get the fitting algorithm over all data points will lead to algorithms of different computational complexity. We summarize the computational properties of the four distance measures discussed below.

Algorithm	distance computation	minimization
Algebraic	closed form	closed form
3L	closed form	closed form
Taubin	closed form	iterative
Euclidean	closed or iterative	iterative

Table 1: Computational properties of the four distance measures: algebraic distance, 3L distance, Taubin's approximation, and Euclidean distance

Given the doubly iterative nature of the Euclidean form, many researchers have avoided it. However, we show below that the time difference is not actually so bad as may be expected.

3 Estimation of the Euclidean distance

The Euclidean distance $\mathrm{dist}(\vec{x}_p, \mathcal{Z}(f))$ between a given point $\vec{x}_p$ and the zero set $\mathcal{Z}(f)$ is the minimal distance between $\vec{x}_p$ and the point $\vec{x}_t$ in the zero set whose tangent is orthogonal to the line joining $\vec{x}_p$ and $\vec{x}_t$. To estimate the Euclidean distance we can differentiate between curves and surfaces for which a closed form expression exist and those where an iterative algorithm must be carried out.

3.1 Closed-form expression

For primitive curves and surfaces like straight lines, ellipses, planes, cylinders, and cones, there is a closed form expression for the Euclidean distance from a point to the zero set and we use these. While the closed form expression for a straight line and a plane is trivial, the closed form expression for the other primitive curves and surfaces is slightly complicated. Below, we will give an idea how the Euclidean distance can be estimated in a closed form for a straight line, a circle, an ellipse, a plane, a sphere, an elliptical (or circular) cylinder, and an elliptical (or circular) cone.

It is a well-known that most fitting methods are not pose invariant (see Sec. 4.4). To improve the pose invariance of the fittings we *normalize* the data set in terms of general moments to affine transformations at first. For a detailed description on how to normalize the data set we refer to the appropriate, numerous literature (e.g. [Galvez and Canton 1993, Hartley and Zissermann 2000, Rothe *et al.* 1996, Wang 1977]). Applying the normalization we get the data set in a so-called *standard position* with respect to a given transformation, here the affine transformation. The standard position can be understood as a special representative of the equivalence class corresponding to the transformation. If the data set is in standard position, a) it is ensured that we get more stable fitting results for all methods, and b) the framework to estimate the Euclidean distance can be simplified for all axe-symmetrical curves and surfaces substantially. In particular, we normalize by translating the data to the centroid, rotating the principal axes of the data to align with the coordinate axes, and applying anisotropic scaling to fit in the unit cube.

3.1.1 Straight line

The Euclidean distance between a point $\vec{x}_p$ and an arbitrary straight line $\mathcal{Z}(f) = a_{10}x + a_{01}y + a_{00}$ can be directly estimated by

$$\mathrm{dist}_E\left(\vec{x}_p, \mathcal{Z}(f)\right) = \left(a_{10}x_p + a_{01}y_p + a_{00}\right)/N \tag{11}$$

where $N = \sqrt{a_{10}^2 + a_{01}^2}$.

3.1.2 Circle

The Euclidean distance between a point $\vec{x}_p$ and an arbitrary circle $\mathcal{Z}(f) = a_{20}(x^2 + y^2) + a_{10}x + a_{01}y + a_{00}$ can be directly estimated by

$$\mathrm{dist}_E\left(\vec{x}_p, \mathcal{Z}(f)\right) = \|\vec{x}_p - \vec{x}_m\| - r\ . \tag{12}$$

Center $\vec{x}_m$ and radius r can be deduced from the $\{a_{rs}\}$ as

$$\vec{x}_m = \left[-\frac{a_{10}}{2a_{20}}, -\frac{a_{01}}{2a_{20}}\right]^T,\ r^2 = \left(\frac{a_{10}}{2a_{20}}\right)^2 + \left(\frac{a_{01}}{2a_{20}}\right)^2 - \frac{a_{00}}{a_{20}}\ . \tag{13}$$

3.1.3 Ellipse

The Euclidean distance between a point $\vec{x}_p$ and the ellipse $\mathcal{Z}(f) = a_{20}x^2 + a_{02}y^2 + a_{11}xy + a_{10}x + a_{01}y + a_{00}$ cannot estimated by a single expression. First, we need an expression for the distance

$$\sqrt{(x_p - x)^2 + (y_p - y)^2} \rightarrow \ \mathrm{Min},\ \ \vec{x} \in \mathcal{Z}(f)\ . \tag{14}$$

Instead of transforming the ellipse into a normal position we can transform $\vec{x}_p$ into the coordinate system of the ellipse $\vec{x}_p \rightarrow \vec{x}_q$.

$$\begin{array}{rcl} x_q &=& (x_p - x_c)\cos(\varphi) \;+\; (y_p - y_c)\sin(\varphi) \\ y_q &=& -(x_p - x_c)\sin(\varphi) \;+\; (y_p - y_c)\cos(\varphi) \end{array} \tag{15}$$

with $\vec{x}_c$ as the center and φ the angle of the ellipse with respect to the x-axis. The parameters $\vec{x}_c$, A, B, and φ (cf. Fig. 1) can be directly deduced from the $\{a_{rs}\}$.

$$\begin{array}{l} \vec{x}_c = \left[\; \dfrac{a_{11}a_{01} - 2a_{02}a_{10}}{4a_{20}a_{02} - a_{11}^2} \quad , \quad \dfrac{a_{11}a_{10} - 2a_{20}a_{01}}{4a_{20}a_{02} - a_{11}^2} \; \right]^T \\ \\ A = \sqrt{h/\lambda_1}\,, \; B = \sqrt{h/\lambda_2} \\ \quad \lambda_{1,2} = \; \frac{1}{2}\left(a_{20} + a_{02} \pm \sqrt{(a_{20} - a_{02})^2 + a_{11}^2} \right) \\ \quad h = \quad -\left(a_{20}x_c^2 + a_{02}y_c^2 + a_{11}x_c y_c + a_{10}x_c + a_{01}y_c + a_{00} \right) \\ \varphi = \arctan\left(\dfrac{a_{02} - a_{20} + \sqrt{(a_{20} - a_{02})^2 + a_{11}^2}}{a_{11}} \right) \end{array} \tag{16}$$

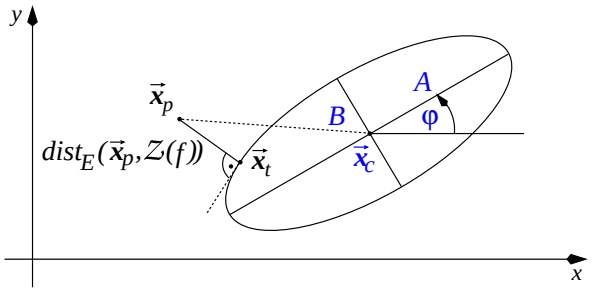


Figure 1: Euclidean distance from $\vec{x}_p$ to the ellipse. $\vec{x}_t$ is the nearest point to $\vec{x}_p$.

Now, we have to solve

$$(x_q - x)^2 + (y_q - y)^2 \rightarrow \text{ Min} \tag{17}$$

with $(x/A)^2 + (y/B)^2 = 1$. There we using a nice, but probably little known algorithm, published in e.g. [Voss and Süße, 1995]. Then, Eq. (17) can be expressed in terms of polar angles by $x = A \cdot \cos\psi$ and $y = B \cdot \sin\psi$:

$$f(\psi) = (x_q - A \cdot \cos\psi)^2 + (y_q - B \cdot \sin\psi)^2 \rightarrow \text{ Min}\,. \tag{18}$$

Applying $\partial f(\psi)/\partial\psi = 0$ we get the goniometrical equation

$$A \cdot x_q \cdot \sin\psi - B \cdot y_q \cdot \cos\psi - (A^2 - B^2) \cdot \cos\psi \cdot \sin\psi = 0\,. \tag{19}$$

To estimate ψ we substitute $t = \tan(\psi/2)$, and we get with

$$\sin\psi = \frac{2t}{1+t^2} \quad \text{and} \quad \cos\psi = \frac{1-t^2}{1+t^2} \tag{20}$$

Eq. (21), $t \neq 0$,

$$By_q \cdot t^4 + 2(A^2 - B^2 + Ax_q) \cdot t^3 - 2(A^2 - B^2 - Ax_q) \cdot t - By_q = 0\,. \tag{21}$$

From the resulting solutions one is selected, for which the distance takes a minimum. The appropriate point $\vec{x}_r$ has to be finally transformed into the original coordinate system to get point $\vec{x}_t$ of the ellipse nearest to $\vec{x}_p$ (cf. Fig. 1)

$$\begin{array}{rcccccl} x_t & = & x_r\cos(\varphi) & - & y_r\sin(\varphi) & + & x_c \\ y_t & = & x_r\sin(\varphi) & + & y_r\cos(\varphi) & + & y_c\,. \end{array} \tag{22}$$

Finally, we get the Euclidean distance as

$$\mathrm{dist}_E\left(\vec{x}_p, \mathcal{Z}(f)\right) = \|\vec{x}_p - \vec{x}_t\|. \tag{23}$$

3.1.4 Plane

The Euclidean distance between a point $\vec{x}_p$ and an arbitrary plane $\mathcal{Z}(f) = a_{100}x + a_{010}y + a_{001}z + a_{000}$ can be directly estimated by

$$\mathrm{dist}_E\left(\vec{x}_p, \mathcal{Z}(f)\right) = \left(a_{100}x_p + a_{010}y_p + a_{001}z_p + a_{000}\right)/N \tag{24}$$

where $N = \sqrt{a_{100}^2 + a_{010}^2 + a_{001}^2}$.

3.1.5 Sphere

The Euclidean distance between a point $\vec{x}_p$ and an arbitrary sphere $\mathcal{Z}(f) = a_{200}(x^2 + y^2 + z^2) + a_{100}x + a_{010}y + a_{001}z + a_{000}$ can be directly estimated by

$$\mathrm{dist}_E\left(\vec{x}_p, \mathcal{Z}(f)\right) = \|\vec{x_m} - \vec{x}_p\| - r\,. \tag{25}$$

Center $\vec{x}_m$ and radius r can be deduced from the $\{a_{rs}\}$ as

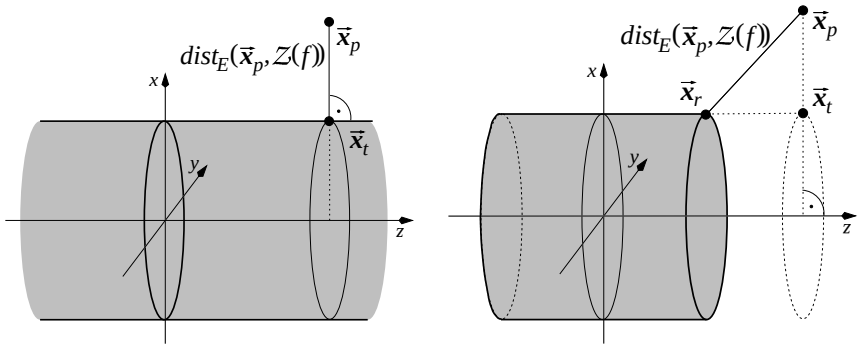
$$\begin{aligned} \vec{x}_m &= \left[-\frac{a_{100}}{2a_{200}}, -\frac{a_{010}}{2a_{200}}, -\frac{a_{001}}{2a_{200}}\right]^T \\ r^2 &= \left(\frac{a_{100}}{2a_{200}}\right)^2 + \left(\frac{a_{010}}{2a_{200}}\right)^2 + \left(\frac{a_{001}}{2a_{200}}\right)^2 - \frac{a_{000}}{a_{200}}\,. \end{aligned} \tag{26}$$

3.1.6 Elliptical cylinder

If the generatix g of the elliptical cylinder is parallel to one axis, e.g. here the z axis, any arbitrary normalized elliptical cylinder can be expressed by the implicit polynomial $\mathcal{Z}(f) = a_{200}x^2 + a_{020}y^2 + a_{110}xy + a_{100}x + a_{010}y + a_{001}z + a_{000}$ with $a_{200}, a_{020} > 0$. The estimation of the Euclidean distance can be reduced to the estimation of the Euclidean distance to the directrix d, here an ellipse $\mathcal{Z}(d) = a_{200}x^2 + a_{020}y^2 + a_{110}xy + a_{100}x + a_{010}y + a_{000}$, $a_{200}, a_{020} > 0$. Thus, to estimate the Euclidean distance for an elliptical cylinder we only have to estimate the Euclidean distance to an ellipse (see Sec. 3.1.3).

If the Euclidean distance is given, we have to consider if the estimated $\vec{x}_t$ belongs to the cylinder element or not, which means if the generatrix g of the elliptical cylinder is infinite in length or not. In case of a finite generatrix we can simply estimate the point $\vec{x}_r \in \mathcal{Z}(f)$ nearest to $\vec{x}_t$ (see Fig. 2b.).

$$\mathrm{dist}_E\left(\vec{x}_p, \mathcal{Z}(f)\right) = \begin{cases} \|\vec{x}_p - \vec{x}_t\| & \text{if } g \to \infty \\ \|\vec{x}_p - \vec{x}_r\| & \text{else} \end{cases} \tag{27}$$



a. (Infinite) generatrix g b. Finite generatrix g

Figure 2: Estimation of $\mathrm{dist}_E(\vec{x}_p, \mathcal{Z}(f))$ for an elliptical cylinder. $\vec{x}_t$ is the nearest point to $\vec{x}_p$ on the (infinite) cylinder. If $\vec{x}_t$ does not belong to the cylinder element, $\vec{x}_r$ can be used.

3.1.7 Elliptical cone

If the generatix g of the elliptical cone is parallel to one axis, e.g. here the z axis, any arbitrary normalized elliptical cone can be expressed by the implicit polynomial $\mathcal{Z}(f) = a_{200}x^2 + a_{020}y^2 + a_{002}z^2 + a_{110}xy + a_{100}x + a_{010}y + a_{001}z$ with $a_{200}, a_{020} > 0$ and $a_{002} < 0$. From the $\{a_{ijk}\}$ the fixed point $\vec{x}_0$ (vertex of the cone) can be directly deduced.

$$\vec{x}_0 = -\begin{pmatrix} a_{200} & \frac{1}{2}a_{110} & 0 \\ \frac{1}{2}a_{110} & a_{020} & 0 \\ 0 & 0 & a_{002} \end{pmatrix}^{-1} \cdot \begin{pmatrix} a_{100} \\ a_{010} \\ a_{001} \end{pmatrix} \tag{28}$$

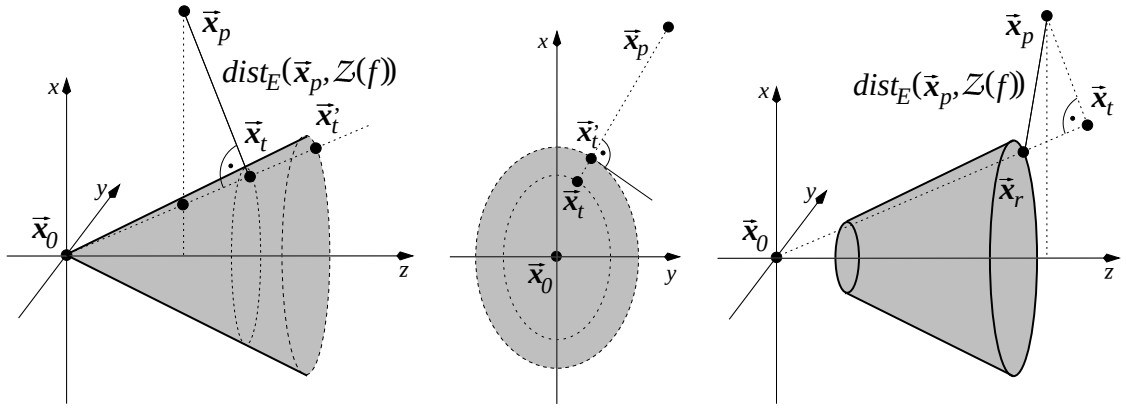
The estimation of the Euclidean distance can be done in two steps. First the estimation of the Euclidean distance can be reduced to the estimation of the Euclidean distance to the directrix d, here an ellipse $\mathcal{Z}(d) = a'_{200}x^2 + a'_{020}y^2 + a'_{110}xy + a'_{100}x + a'_{010}y + a'_{000}$, $a'_{200}, a'_{020} > 0$. Now, the parameters A and B (axis of the directrix d) as well as φ can be deduced from the $\{a'_{ijk}\}$ (see Sec. 3.1.3).

Note, this step is similar to that in the previous section, but $\{a'_{ijk}\}$ are different to $\{a_{ijk}\}$. In the second step the $\vec{x}_t$ with $(\vec{x}_o - \vec{x}_t) \perp (\vec{x}_p - \vec{x}_t)$ is determined, which yields the Euclidean distance. The only problem left is how to estimate $\vec{x}_t$. Therefore, we estimate $\vec{x}'_t$ located on the directrix d of the cone (see Fig. 3b.). The respecting $\vec{x}_t$ can be estimated using simple geometric relations (see Fig. 3a.). Then, the estimation of the Euclidean distance is trivial using Eq. 27. In the case that the estimated $\vec{x}_t$ does not belong to the cone element (see Fig. 3c.), the reasoning is similar to that in the previous section.

3.2 Iterative estimation

If no closed form expression is known for the Euclidean distance (e.g. for common shapes like superellipse and superellipsoid, torus and supertoroid), an iterative optimization procedure must be carried out. We use the following simple iterative algorithm (Fig. 4):

1. The estimation is initialized by intersecting the curve or surface with the straight line defined by the center point $\vec{x}_c$ and the point $\vec{x}_p$. As result we get the initial point $\vec{x}_t^{[0]}$. By the initial solution, the upper bound for the distance is determined.



a. (Infinite) generatrix g b. Normalized cone c. Finite generatrix g

Figure 3: Estimation of $\mathrm{dist}_E(\vec{x}_p, \mathcal{Z}(f))$ for an elliptical cone. $\vec{x}_t$ is the nearest point to $\vec{x}_p$ on the (infinite) cone. If $\vec{x}_t$ does not belong to the cone element, $\vec{x}_r$ can be used.

2. In the second step the estimate is updated by a modified steepest descent method (cf. Fig. 4b.):

$$\vec{x}_t^{[s+1]} = \vec{x}_t^{[s]} + \alpha^{[s]} \nabla f\left(\vec{x}_t^{[s]}\right), \; s = 0, 1, 2, \ldots \; . \tag{29}$$

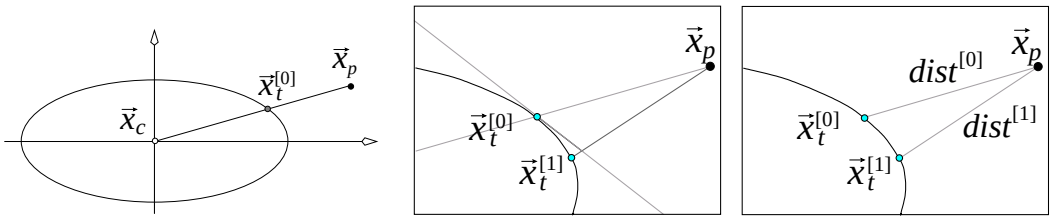
3. We have to decide by an objective function $\mathcal{F}$, if $\vec{x}_t^{[s+1]}$ will be accepted as new solution. The objective function is formulated as

$$\mathcal{F}(\vec{x}_p, \vec{x}_t^{[s+1]}, \mathcal{Z}(f)) = \min\left(\mathrm{dist}_E(\vec{x}_p, \vec{x}_t^{[s]}), \mathrm{dist}_E(\vec{x}_p, \vec{x}_t^{[s+1]})\right). \tag{30}$$

If $dist_E(\vec{x}_p, \vec{x}_t^{[s+1]}) < dist_E(\vec{x}_p, \vec{x}_t^{[s]})$, we accept $\vec{x}_t^{[s+1]}$ as new (local) solution and the algorithm will be continued with step 2 until

$$\left| dist_E(\vec{x}_p, \vec{x}_t^{[s+1]}) - dist_E(\vec{x}_p, \vec{x}_t^{[s]}) \right| < thrs. \tag{31}$$

Otherwise $\vec{x}_t^{[s+1]} := \vec{x}_t^{[s]}$, $\alpha^{[s+1]} = -\tau\alpha^{[s]}$, $\tau > 0$ and the algorithm will be continued with step 2*. To speed up the estimation a criterion to terminate the updating may be used like e.g. $\|\vec{x}_t^{[s+1]} - \vec{x}_t^{[s]}\| \leq \varepsilon$, or $s \geq \tau_s$.



a. Initialization b. Updating

Figure 4: Steps to estimate $\mathrm{dist}_E(\vec{x}_p, \mathcal{Z}(f))$ of $\vec{x}_p$ to the zero set $\mathcal{Z}(f)$ of an ellipse

3.3 Estimation error

Given the Euclidean distance error for each point, we then compute the curve or surface fitting error. The standard least-squares method tries to minimize

$$\sum_p \mathrm{dist}_E^2(\vec{x}_p, \mathcal{Z}(f)), \tag{32}$$

which is unstable if there are outliers in the data. Outlying data can give so strong an effect in the minimizing that the parameters are distorted. Replacing the squared residuals by another function $\rho(\mathrm{dist}_E(\vec{x}_p, \mathcal{Z}(f)))$ can reduce the effect of outliers. Appropriate minimization criteria including functions were discussed in for instance [Besl 1990, Zhang 1997]. It seems difficult to select a function which is generally suitable. Following the results given in [Rey 1983] the best choice may be the so-called $\mathcal{L}_\nu$ (*least power*) function

$$\mathcal{L}_\nu := |\mathrm{dist}_E(\vec{x}_p, \mathcal{Z}(f))|^\nu / \nu. \tag{33}$$

This function represents a family of functions including the two commonly used functions $\mathcal{L}_1$ (*absolute power*) with $\nu = 1$ and $\mathcal{L}_2$ (*least squares*) with $\nu = 2$. Note, the smaller ν, the smaller is the influence of large errors. The associated so-called *influence function* is the first derivative of ρ. It is obvious that $\mathcal{L}_1$ penalizes outliers to a lesser extent than $\mathcal{L}_2$, and so the selection of the former would be more appropriate if numerous outliers are present in the data. But, the $\mathcal{L}_2$ is familiar as the Euclidean distance. A good compromise may be to start with the least power estimator $\mathcal{L}_\nu$ with $\nu \approx 1.2$, for which a good error estimation may be expected [Rey 1983]. After a number of iterations, e.g. $s = 3$, we remove the 10% worst data points and continue the fitting with the commonly used $\mathcal{L}_2$ estimator. Note, one can use a histogram analysis to improve the reliability of data point removal.

4 Experimental results

In the previous section we described the Euclidean fitting based on the estimated Euclidean distance between a 2D point or a 3D point and a curve or a surface. In this section we summarize empirical testing of the proposed fitting method in terms of efficiency, correctness and robustness. We only evaluate surfaces because they are the more complex (more parameters) and computational expensive case. The 3D data were generated by adding isotropic Gaussian noise $\sigma = \{1\%, 5\%, 10\%, 20\%\}$. Additionally the surfaces were partially occluded. The visible surfaces were varied between $1/2$ (maximal case) and $1/6$ of the full 3D cylinder. The performance of Euclidean fitting (*EF*) is compared with Algebraic fitting (*AF*), 3L fitting (*LF*), and Taubin's fitting (*TF*). Finally, we look to the pose invariance of the fitting methods. For all experiments we include in the four fitting methods the same constraints which describe the expected surface type to enforce the fitting of a special surface type (cmp. Tab. 3).

4.1 Efficiency

A good fitting algorithm has to be as efficient as possible in terms of run time and formal complexity. While the problem of computational cost is no longer a really hard problem because of the rapidly increasing machine speed, we should guarantee the fitting has acceptable computational cost as well as relatively low complexity.

The algorithms have been implemented in C and the computation was performed on a Pentium III 466 MHz. The average computational costs for the four algorithms are in Tab. 2.

Table 2: Average computational costs in milliseconds per 1000 points. Note, the average computational costs for the *LF* are on the assumption that the pair of data sets is available. The cost of generating these might be quite substantial because of the need of an Euclidean distance transformation.

	AF	*LF*	*TF*	*EF*
Plane	0.958	0.967	1.042	2.417
Sphere	1.208	1.301	1.250	3.208
Circular cylinder	3.583	3.462	3.625	12.375
Elliptical cylinder	13.292	12.864	13.958	241.667
Circular cone	15.667	15.445	15.833	288.375
Elliptical cone	15.042	15.465	15.375	291.958
General quadric	18.208	18.646	18.458	351.083

As expected the *AF*, *LF*† and *TF* have the best performance. The *EF* algorithm requires a repeated search for the point x_t closest to x_p and the calculation of the Euclidean distance. A quick review of the values in Tab. 2 shows that the computational costs increase if we fit an elliptical cylinder, a circular or an elliptical cone respectively a general quadric. The algorithm to estimate the distance by the closed form solution respectively the iterative algorithm is more complicated in these cases (cf. Sec. 3). The number of necessary iterations to estimate the $\{a_{ijk}\}$ using *TF* and *EF* is also influenced by the required precision of the LM algorithm to terminate the updating process. In summary the efficiency of *EF* is bounded by a factor of about 20 times the performance of the other algorithms but is still computationally reasonable for up to 10^6 data points if real-time performance is not needed.

4.2 Correctness

It is obvious that the fitting result should describe the data set by the correct curve or surface type. That means that it should not fit a false type to the data. The problem that we have to deal with is to decide if a fitting is wrong or not independent from our expectations. But, correctness makes only sense if we fit a special curve or surface type to the data. Otherwise, if we fit a general curve or surface to the data, every result is correct.

To verify the correctness we tested if the fitting result of the (constrained) eigenvalue analysis corresponds to the general curve or surface invariants (see Tab. 3). If one solution satisfies the conditions for one curve or surface type, it is assumed that the fitting is correct in sense of an interpretable real curve or surface. Otherwise, the fitting will be defined as failure. The used invariants for the curves (cf. Eq. 34) and surfaces (cf. Eq. 35) are the determinant $\triangle$ of the parameter matrix, the subdeterminant σ expanded by the absolute parameter, the trace S of the submatrix, and for the 2D surfaces additionally the sum T of all subdeterminants of σ expanded by the elements of the primary diagonal (see Tab.3).

$$\triangle = \begin{vmatrix} a_{20} & a_{11} & a_{10} \\ a_{11} & a_{02} & a_{01} \\ a_{10} & a_{01} & a_{00} \end{vmatrix} \quad \sigma = \begin{vmatrix} a_{20} & a_{11} \\ a_{11} & a_{02} \end{vmatrix} \quad S = a_{20} + a_{02}. \tag{34}$$

$$\triangle = \begin{vmatrix} a_{200} & a_{110} & a_{101} & a_{100} \\ a_{110} & a_{020} & a_{011} & a_{010} \\ a_{101} & a_{011} & a_{002} & a_{001} \\ a_{100} & a_{010} & a_{001} & a_{000} \end{vmatrix} \quad \sigma = \begin{vmatrix} a_{200} & a_{110} & a_{101} \\ a_{110} & a_{020} & a_{011} \\ a_{101} & a_{011} & a_{002} \end{vmatrix} \tag{35}$$

$S = a_{200} + a_{020} + a_{002}$
$T = a_{200}a_{020} + a_{200}a_{002} + a_{020}a_{002} - a_{110}^2 - a_{101}^2 - a_{011}^2.$

Table 3: Invariants for 2D curves and surfaces.

		Invariants
2D curve	Ellipse	$\triangle \neq 0, \sigma > 0, \triangle \cdot S < 0$
2D surface	Cylinder	$\triangle = 0, T > 0$
	Cone	$\triangle = 0, S \cdot \sigma$ and T not both > 0
	Ellipsoid	$\triangle < 0, S \cdot \sigma > 0, T > 0$

In our experiments *AF* and *LF* failed sometimes (up to 23 percent) respecting our expectations, especially in case of $\sigma > 10\%$ and a sparse data set or if most of the 3D object is occluded. For *TF* and *EF* we had no failures in our experiments.

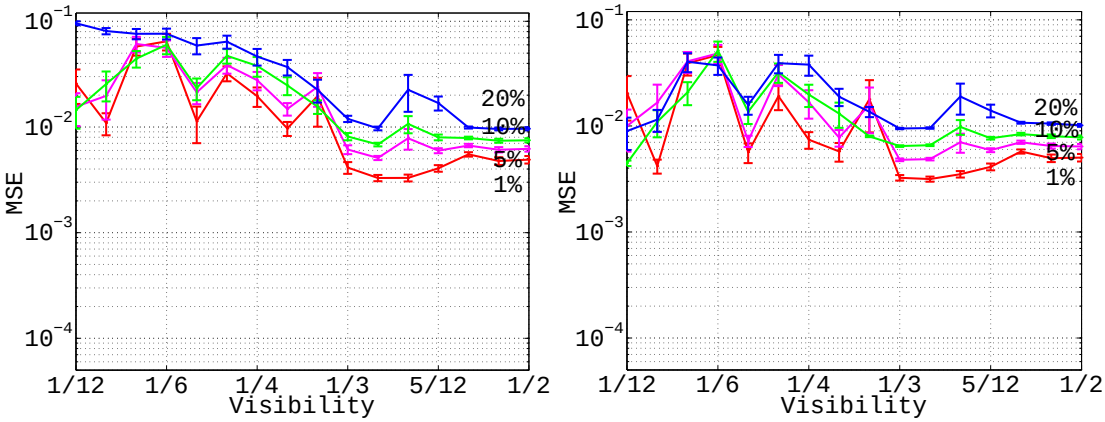
4.3 Robustness

A fitting method must degrade gracefully with increasing noise in the data, with a decrease in the available relevant data, and with an increase in the irrelevant data. To evaluate the robustness of the proposed *EF*, we use synthetic generated data describing an elliptical cylinder with the axes $A = 60$, $B = 40$ and the generatrix $g = 100$. The 3D data were generated by adding isotropic Gaussian noise $\sigma = \{1\%, 5\%, 10\%, 20\%\}$ and partially occlusion exposing between $1/2$ (maximal case) and $1/6$ of the full 3D cylinder. In the first experiment the number of 3D points for the simulated cylinder was $n = 180$ (maximal case) and to measure the average fitting error each experiment runs 500 times. The reported error is the Euclidean geometric distance between the 3D data points and the estimated surfaces. The mean squares errors (*MSE*'s) and standard deviations of the different fittings are in Fig. 5.

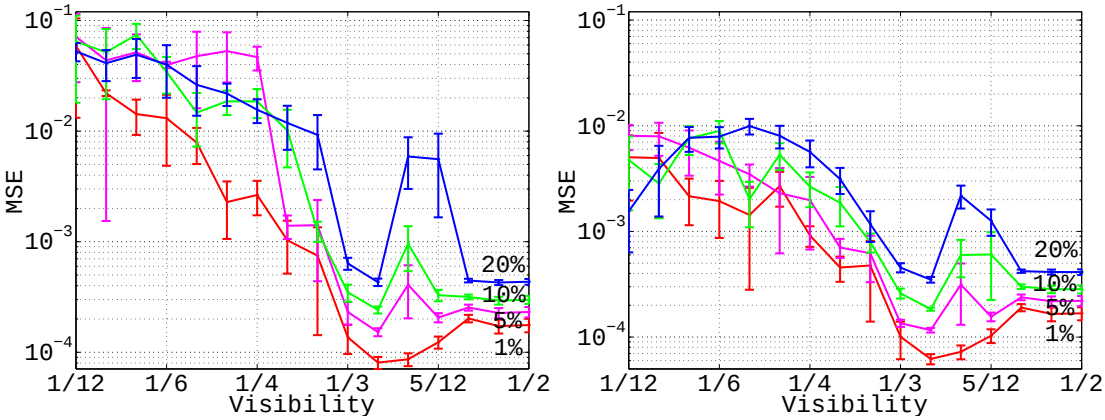
As expected, *TF* and *EF* yield the best result with respect to the mean and standard deviation, and the mean for *EF* is always lower than for the other three algorithms. The results of *AF* and *LF* are only partially acceptable, because of the mean and the standard deviation. In the direct comparison of *TF* with *EF* the results of *EF* are much better and the mean of *EF* is always lower than the mean of the other three algorithms. As mentioned in Sec.4.2, *AF* and *LF* can sometimes give wrong results which means that the fitted curve or surface typs does not come up with our expectations. We removed all failed fittings out of the considerations.

In the second experiment, the number of 3D points was stepwise descreased from $n = 3000$ down to $n = 10$ 3D data points to evaluate the behaviour of the several fitting methods. Each experiment runs 500 times. The mean squares errors (*MSE*'s) and standard deviations of the different fittings are in Fig. 6.

As expected, *TF* and *EF* yield also the best results in this experiment. With descreased point density especially the *AF* and *LF* becomes more and more unstable which is reflected in the mean respectively the standard deviation. *TF* and *EF* is very stable even with only $n = 10$ 3D data points.



a. Algebraic fitting. b. 3L fitting.



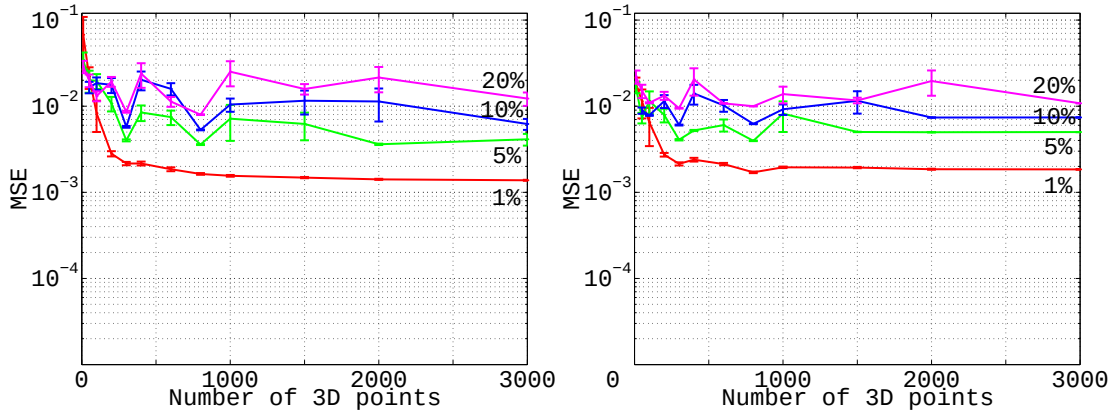
c. Taubin's fitting. d. Euclidean fitting.

Figure 5: Average least squares error with standard deviation error bars fitting a synthetic generated cylinder (axes $A = 60$, $B = 40$, generatrix $g = 100$) with added Gaussian noise $\sigma = \{1\%, 5\%, 10\%, 20\%\}$. The visible surfaces were observed between $1/2$ (maximal case) and $1/12$ of the full 3D cylinder. The number of 3D points was $n = 180$. The number of trials was 500.

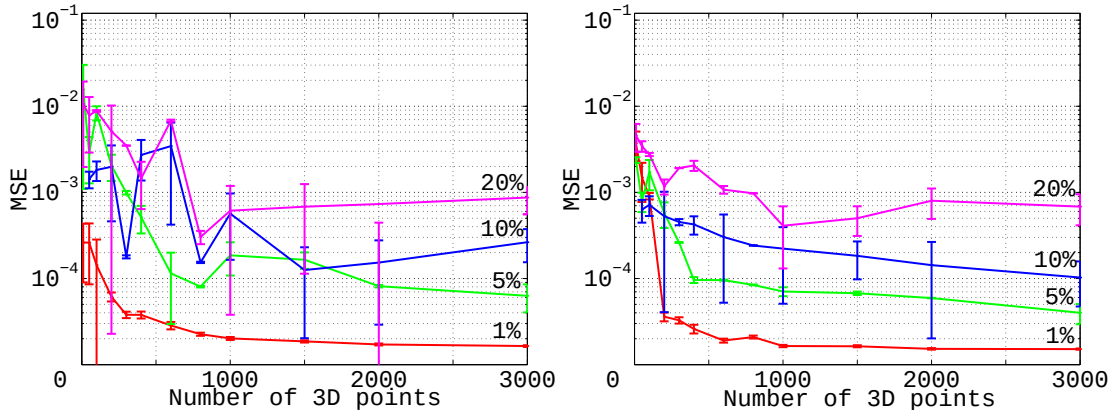
Reviewing some visualized fitting results for synthetic data sets (see Fig. 7) we can assume that the results of *EF* are much better in the sense of robustness and the expected surface than the results of *AF*, *LF* and *TF*. While the robustness (see Fig. 5) of *TF* is better than *AF* and *LF*, the fitted surfaces of all three fitting methods are mostly larger than the optimal surface. The results of *EF* are both robust and they correspond much more to our expectations.

4.4 Pose invariance

It is obvious that the fitting results should be pose invariant. But, it is well known that this reasonable and necessary requirement cannot be always guaranted by all four viewed fitting methods. To evaluate the pose invariance we use a real data set (see Fig. 8a.) describing an elliptical cylinder. The normalized data set was a) shifted, b) rotated, and c) both rotated and shifted. A quick review of the residuals (*MSE*) in Tab. 4 shows that the *AF*, the *LF* as well as the *TF* are not pose invariant while the *EF* is pose invariant. To illustrate the pose dependency, the fitting results for position 3 are visualized in Fig. 8.



a. Algebraic fitting. b. 3L fitting.



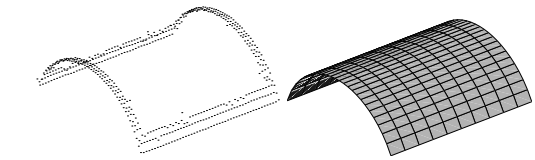
c. Taubin's fitting. d. Euclidean fitting.

Figure 6: Average least squares error with standard deviation error bars fitting a synthetic generated cylinder (axes $A = 60$, $B = 40$, generatrix $g = 100$) with added Gaussian noise $\sigma = \{1\%, 5\%, 10\%, 20\%\}$. The number of 3D points was stepwise descresed from $n = 3000$ down to $n = 10$. The visible surfaces was $1/3$ of the full 3D cylinder. The number of trials was 500.

Table 4: Residuals fitting an elliptical cylinder. The normalized cylinder was shifted by $t = [0.3, 0.2, 0.1]$ (pos. 1), rotated by $\vartheta = \pi/12$ and $n = [0.5, 1.0, 0.5]$ (pos. 2), shifted and rotated (pos. 3).

		normal pos	pos. 1	pos. 2	pos. 3
AF	$[10^{-3}]$	0.5242	2.0181	2.6950	1.8271
LF	$[10^{-3}]$	0.5448	0.9629	4.0068	2.5432
TF	$[10^{-3}]$	0.5024	1.5143	2.0277	1.3817
EF	$[10^{-3}]$	0.4021	0.4152	0.8634	0.6088

As we can see *AF* yields an elliptical cylinder with correct direction but too large axes, while the results of *TF* is an elliptical cylinder with a correct direction but too small axes. The *LF* results give a cylinder with nearly correct axes but slightly incorrect direction vector. The result of *EF* describes the data set best (cmp. Tab. 4).



Synth. data. True shape.

$n = 100$		$\sigma = 5\%$	
$\sigma = 1\%$	$\sigma = 10\%$	$n = 500$	$n = 10$

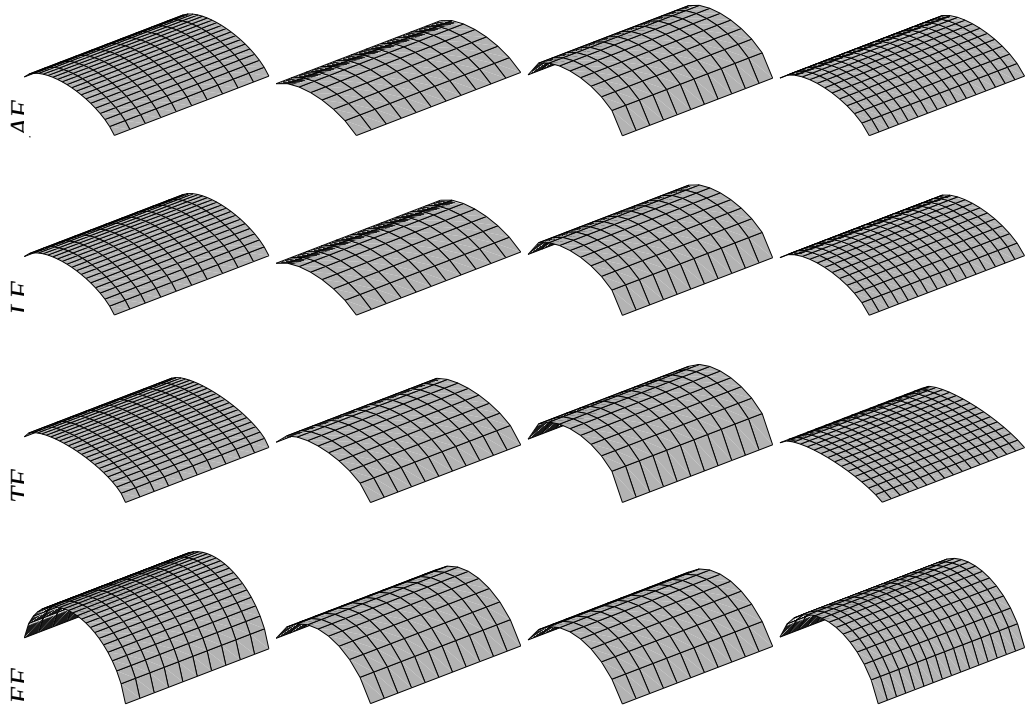
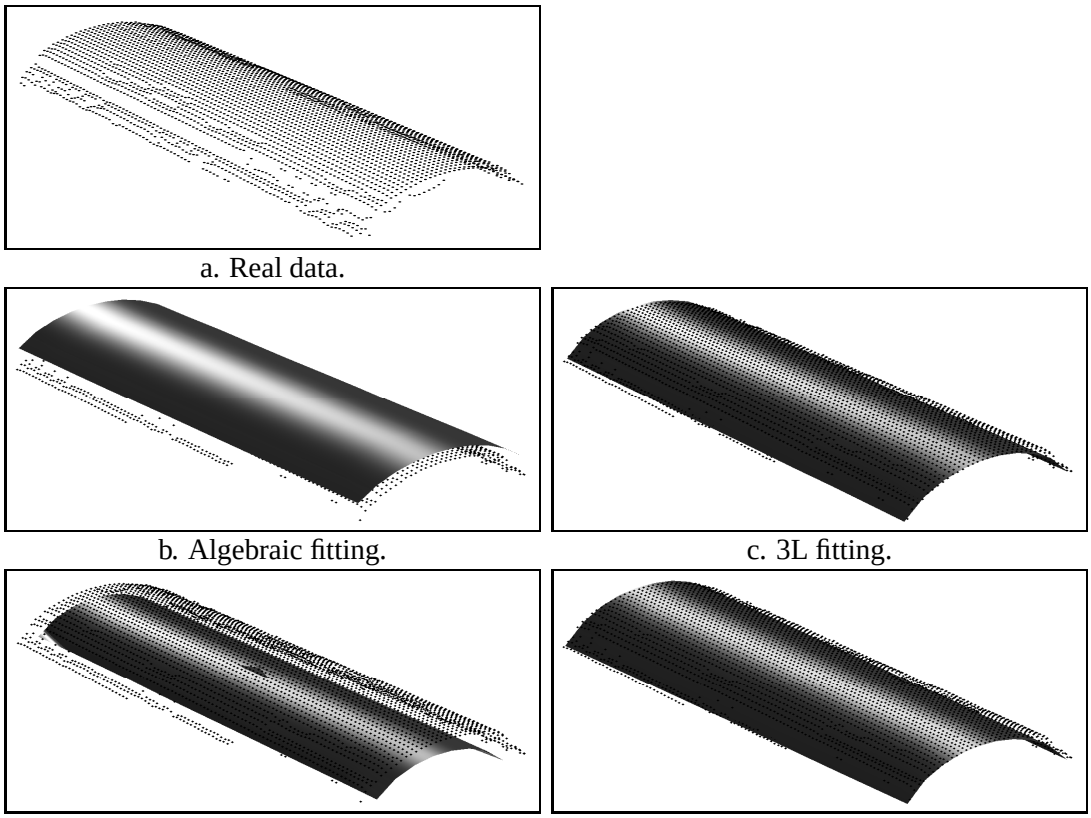


Figure 7: Fitting results for a synthetic generated data set. The noise level was zero and the visible surface $1/6$ of the full 3D cylinder. Note, the maximal visible surface is $1/2$ of the full 3D cylinder.



a. Real data.

b. Algebraic fitting.

c. 3L fitting.

d. Taubin's fitting.

d. Euclidean fitting.

Figure 8: Fitting results for a real range data (points). The normalized data set was shifted by $t = [0.3, 0.2, 0.1]$ and rotated by $\vartheta = \pi/18$ and $n = [0.5, 1.0, 0.5]$.

5 Conclusion

We revisited the Euclidean fitting of curves and surfaces to 3D data to investigate if it is worth considering Euclidean fitting again. The focus was on the quality and robustness of Euclidean fitting compared with the commonly used Algebraic and Taubin's fitting and also the 3L fitting. Now, we can conclude that robustness and accuracy increases sufficiently compared to the other methods and Euclidean fitting is more stable with increased noise, as well as invariant to Euclidean transformations.

The main disadvantage of the Euclidean fitting, computational cost, has become less important due to rising computing speed. In our experiments the computational costs of Euclidean fitting were only about 2-19 times worse than Taubin's fitting. This relation probably cannot be improved substantially in favor of Euclidean fitting, but the absolute computational costs are becoming an insignificant deterrent to usage, especially if high accuracy is required.

The work was funded by the CAMERA (*CA*d *M*odelling of Built *E*nvironments from *R*ange *A*nalysis) project, an EC TMR network (ERB FMRX-CT97-0127).

References

[Albano 1974] Albano, A. Representation of digitized contours in terms of conic arcs and straight-line segments. *CGIP*, 3:23, 1974.

[Allen 1935] Allan, F. E. The general form of the orthogonal polynomial for simple series with proofs of their simple properties. In *Proc. Royal Soc. Edinburgh*, pp. 310–320, 1935.

[Besl 1990] Besl, P. J. *Analysis and Interpretation of Range Images*, chap. Geometric Signal Processing. Springer, Berlin-Heidelberg-New York, 1990.

[Besl and Jain 1985] Besl, P. J. and R. C. Jain. Three-dimensional object recognition. *Computing Survey*, 17(1):75–145, 1985.

[Blane *et al.* 2000] Blane, M. M. and Z. Lai and H. Çivi and D. B. Cooper. The 3L algorithm for fitting implicit polynomial curves and surfaces to data. *IEEE Trans. on PAMI*, 22:298–313, 2000.

[Bookstein 1979] Bookstein, F. L. Fitting conic sections to scattered data. *CGIP*, 9:56–71, 1979.

[Chui and Chen 1987] Chui, C. K. and G. Chen. *Kalman filtering with real time applications.* Springer, Berlin-Heidelberg-New York, 1987.

[Degeeter *et al.* 1997] Degeeter, J. and H. van Brussel and J. Deschutter and M. Decreton. *A smoothly constrained Kalman filter*. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(10):1171–1177, 1997.

[Dickmanns and Graefe 1988] Dickmanns, E. D. and V. Graefe. *Dynamic monocular machine vision. Machine Vision and Applications*, 1:223–240, 1988.

[Duda and Haet 1972] Duda, R. O. and P. E. Hart. The use of Hough transform to detect lines and curves in pictures. *Comm. Assoc. Comp. Machine*, 15:11–15, 1972.

[Fitzgibbon and Fisher 1995] Fitzgibbon, A. W. and R. B. Fisher. A buyer's guide to conic fitting. In *6th British Machine Vision Confernce*, pp. 513–522, 1995.

[Galvez and Canton 1993] Galvez, J. M. and M. Canton Normalization and shape recognition of three-dimensional objects by 3D moments. *Pattern Recognition*, 26:667–681, 1993.

[Grimson 1990] Grimson, W. E. L. *Object recognition by computer: The role of geometric constraints.* MIT-Press, Cambridge-London, 1990.

[Hartley and Zissermann 2000] Hartley, R. and A. Zissermann. *Multiple View Geometry in Computer Vision.* Cambridge University Press, 2000.

[Kanatani 1993] Kanatani, K. Renormalization for biased estimation. In *4th Int'l. Conf. Computer Vision*, pp. 599–606, 1993.

[Levenberg 1944] Levenberg, K. A method for the solution of certain nonlinear problems in least squares. *Quarterly of Applied Mathematics*, 2:164–168, 1944.

[Marquardt 1963] Marquardt, D. W. An algorithm for least squares estimation of nonlinear parameters. *Journal of the Soc. of Industrial and Applied Mathematics*, 11:431–441, 1963.

[Paton 1970] Paton, K. A. Conic sections in chromosome analysis. *Pattern Recognition*, 2:39, 1970.

[Rey 1983] Rey, W. J. J. *Introduction to Robust and Quasi-Robust Statistical Methods.* Springer, Berlin-Heidelberg-New York, 1983.

[Rosin 1993] Rosin, P. L. A note on the least square fitting of ellipses. *Pattern Recognition Letters*, 14:799–808, 1993.

[Rothe *et al.* 1996] Rothe, I. and H. Süße and K.Voss. The method of normalization to determine invariants. *IEEE Transactions on Pattern Aanalysis and Machine Intelligence*, 18(4):366–376, 1996.

[Taubin 1991] Taubin, G. Estimation of planar curves, surfaces and non-planar space curves defined by implicit equations, with applications to edge and range image segmentation. *IEEE Transactions on Pattern Aanalysis and Machine Intelligence*, 13(11):1115–1138, 1991.

[Taubin 1993] Taubin, G. An improved algorithm for algebraic curve and surface fitting. In *4th Int'l. Conf. Computer Vision*, pp. 658–665, 1993.

[Voss and Süße, 1995] Voss, K. and H. Süße. *Adaptive Modelle und Invarianten für zweidimensionale Bilder.* Shaker, Aachen, 1995.

[Wang 1977] Wang, K. *Affine-invariant moment method of three-dimensional object identification.* PhD Thesis, Syracuse University, 1977.

[Zhang 1997] Zhang, Z. Parameter estimation techniques: a tutorial with application to conic fitting. *Image and Vision Computing*, 15:59–76, 1997.